Actual Causation in CP-logic*

Joost Vennekens

Lessius Mechelen – Campus De Nayer
J. De Nayerlaan 5, 2860 Sint-Katelijne-Waver

Ever since the seminal work of Lewis [2], the problem of *actual causation* has been extensively studied in philosophy. More recently, this topic has also drawn the attention of the AI community, notably with the work of Halpern and Pearl [1]. In a nutshell, the problem is this: given the causal laws that govern a certain domain, together with a story that has taken place in this domain, when should we say that, in this particular story, some X has caused Y? As a simple example, consider two children, Billy and Suzy, who are considering throwing rocks at a window. The causal laws of this domain are simple: assuming both children will throw accurately, the window will break if at least one of them indeed decides to throw. A story in this domain might be: Suzy decides to throw, but Billy doesn't; Suzy's rock hits the window, which breaks. Clearly, in this story, Suzy was the one who actually caused the window to break.

In order to provide a formal definition of the concept of actual causation, we first of all need a formal representation for its two givens, i.e., for the causal laws of the domain and for the story that has taken place. Following Halpern and Pearl, most contemporary approaches use a structural model for the first part. In the boolean case, such a structural model is an acyclic set of equations of the form $A := \phi$, that define the property A by means of the boolean formula ϕ. The above example requires only one equation:

$$Break := Throws(Suzy) \vee Throws(Billy). \tag{1}$$

When such a structural model is used to represent the causal laws of the domain, the accompanying story is represented by a truth assignment to all the boolean variables, e.g.:

$$Break = true; \qquad Throws(Suzy) = true; \qquad Throws(Billy) = false. \tag{2}$$

Since the story is supposed to have taken place in a domain that obeys the given causal laws, the truth assignment should be consistent with the boolean equations, i.e., in this case, eq. (1) should be satisfied by eq. (2).

Recently, several authors have identified problems with the apprach of Halpern and Pearl. Typically, the source for these problems is sought in the technical details of their definition of actual causation. In this paper, though, I argue that there is also a more fundamental problem, namely, that structural models are not a suitable causal modeling framework in which to address the topic of actual causation. The reason for this is the static nature of the formalism. If we look back at the truth assignment in (2), we see that this is really not a faithfull representation of our earlier story at all: while it correctly states that Suzy threw a rock and that the window is broken, it entirely omits the information that one thing happened *before* the other. Is this missing information relevant? If our goal is to make judgments of actual causation, then it most certainly is: after all, if Suzy had thrown her rock *after* the window was already broken, then she could not possibly have caused the window to break. If Billy and Suzy both throw, this of course becomes particularly pertinent, because then only the first one to hit the window should be counted as an actual cause for its breaking.

Halpern and Pearl recognize this problem in their paper and propose a fix that involves changing equation (1) to include information about which rock will reach the window first. However, this solution is somewhat questionable, since the order in which events happens seems to belong more to the particular story that is being considered than to the causal laws of the domain in general. Moreover, fixing the problem for this particular example does of course not amount to a general solution. In this paper, I argue that what is called for instead is a different formal representation for the causal laws of the domain, which does take into account the dynamic aspects of causation, such as the fact that events always happens in a particular order.

*To appear in *Theory and Practice of Logic Programming*, special issue on the 27th International Conference on Logic Programming (ICLP 2011).

To be more concrete, the langauge I propose is that of *CP-logic* [4]. Following the work of Shafer [3], the semantics of this language uses probability trees to represent the possible ways in which a domain might evolve. Syntactically, a theory in CP-logic consists of a set of causal laws of the form $\epsilon \leftarrow \phi$, that express that ϕ causes ϵ. Here, ϵ could simply be an atomic property, but it could also be a probability distribution over a number of such properties if ϕ has a non-deterministic effect. For instance, if we extend the above example by assuming that the children might miss with a certain probability, we would get:

$$\begin{aligned}(Break : 0.9) &\leftarrow Throws(Suzy)\\ (Break : 0.8) &\leftarrow Throws(Billy)\end{aligned} \tag{3}$$

Under the assumption that both children indeed throw, the semantics of this CP-theory would be given by the following two probability trees (the open circles represent states in which the window is broken):

Both of these probability trees obey the CP-logic theory of (3), in the sense that each transition in this tree corresponds to one of its causal laws. These two trees differ in the order in which the rocks arrive: in the left tree, Suzy's is first, in the right one, Billy's is. Indeed, because the causal laws do not say anything about the relative speed of the throws, both are possible. The probability distribution over the final states (i.e., the leaves of the tree) is the same in both trees, i.e., $P(Break) = 0.98$ in both cases. This is in fact a general property for all theories in CP-logic.

The advantage of CP-logic as a framework for actual causation is that we now do have mathematical objects at our disposal that captures in a natural and direct way all the information that is present in a story, namely, the individual branches of these probability trees. For instance, consider the story that both children throw, that Suzy's rock first reaches the window and breaks it, and afterwards Billy's rock also hits. Essentially, we are then simply saying that we are in the leftmost branch of the left probability tree. In this particular branch, it is easy to make out that it was in fact Suzy's rock that broke the window, not Billy's.

This paper presents a formal definition of actual causation in the context of CP-logic. This definition is based on a criterion of *counterfactual dependency* ("had it not been for X, then Y would not have happened"), which is pretty standard in the literature. However, by itself, this does not suffice. In the above example, for instance, Billy's rock would have broken the window even if Suzy had not thrown, so we would fail to identify hers as the actual cause for the damage. This first idea therefore needs to be amended by saying that this counterfactual dependency should exist in the *relevant* part of the causal model. The nice thing about the formal setting of CP-logic is now that we can decide in very simple yet appealing way which causal laws were actually relevant for causing a certain effect: everything that happened *after* the effect was already present must have been irrelevant. In the probability tree branch discussed above, Billy's rock only hit the window after it was already broken, so this must have been irrelevant. If we therefore ignore this irrelevant event, we find that in the remaining part of our causal model, there does exist a counterfactual dependency between Suzy's throwing and the window's breaking, so Suzy did cause the damage after all. The paper provides formal definitions of these ideas and compares them in more detail to existing approaches.

References

[1] J. Halpern and J. Pearl. Causes and explanations: A structural-model approach. part I: Causes. *The British Journal for the Philosophy of Science*, 56(4), 2005.

[2] D. Lewis. Causation. *journal of philosophy*, 70:556–67, 1973.

[3] G. Shafer. *The art of causal conjecture*. MIT Press, 1996.

[4] J. Vennekens, M. Denecker, and M. Bruynooghe. CP-logic: A language of probabilistic causal laws and its relation to logic programming. *Theory and Practice of Logic Programming*, 9(3):245–308, 2009.